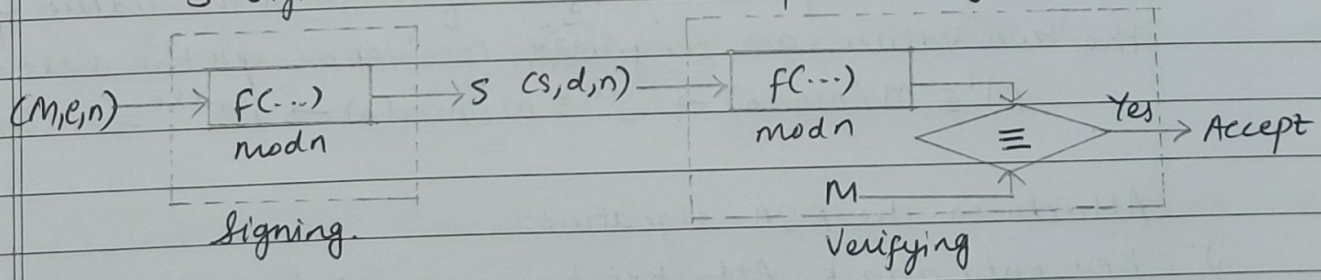
Name: Gargi P Sukhatankar Batch: B1
SAP ID: 60019220125 Branch: CSE(ICB)
Roll No: B027

AC - ASSIGNMENT-2

Q.1. Explain in detail RSA Digital signature scheme along with various attacks on it.

⇒ The RSA cryptosystem idea can also be used for signing and verifying a message. In this case, it is called RSA digital signature scheme. The digital signature changes the roles of the private and public keys. First, the private and public keys of the sender, not the receiver, are used. Second, the sender uses her own private key to sign the document, the receiver uses the sender's public key to verify it.

M: Message (e,n): Sendeo's publickey
S: Signature d: Sender's private key.



General idea behind RSA digital signature scheme.

The signing and verifying sites use the same function, but with different parameters. The verifier compares the message and the output of the function for congruence. If the result is true, the message is accepted.

Key Generation: Key generation in RSA digital signature scheme is exactly the same as key generation in RSA cryptosystem.

$p, q \rightarrow$ two prime numbers.

$n = p \times q$

$\phi(n) = (p-1) * (q-1)$

$e \rightarrow$ public exponent such that $gcd(e, \phi(n)) = 1$

d such that $(e \times d) \bmod \phi(n) = 1$

$d \rightarrow$ private key.

Signing and Verifying:

Signing: Sender creates a signature out of the message and using her private exponent, $S = M^d \bmod n$ and sends the message and signature to the receiver.

Verifying:

Receiver receives M and S. He applies sender's public exponent to the signature to create a copy of the message $M' = S^e \bmod n$. Then he compares the value of M' with the value of M. If the two values are congruent, receiver accepts the message.

$$M' \equiv M \pmod n \rightarrow S^e \equiv M \pmod n \rightarrow M^{d \times e} \equiv M \pmod n$$

Attacks on the RSA signature:

1) Key only attack: Attacker has access only to the sender's public key. She intercepts the pair (M, S) and tries to create another message M' such that $M' \equiv S^e \pmod n$. This problem is as difficult to solve as the discrete logarithm problem. Besides, this is an existential forgery and normally is useless to the attacker.

2) Known Message Attack: Here, the attacker uses the multiplicative property of RSA. Assume that the attacker has intercepted two message - signature pairs (M_1, S_1) and (M_2, S_2) that have been

created using the same private key. If $M = (M_1 \times M_2) \bmod n$ then $S = (S_1 \times S_2) \bmod n$. This is simple to prove because we have

$$S = (S_1 \times S_2) \bmod n = (M_1^d \times M_2^d) \bmod n = (M_1 \times M_2)^d \bmod n = M^d \bmod n$$

Attacker can create $M = (M_1 \times M_2) \bmod n$ and she can create $S = (S_1 \times S_2) \bmod n$ and fool the receiver into believing that S is the real sender's signature on the message M. However, this is an existential forgery as the message M is a multiplication of 2 previous messages created by the sender, not the attacker, M is normally useless.

3) Chosen message attack: This attack also uses the multiplicative property of RSA. Attacker can somehow ask the sender to sign 2 legitimate messages M_1 and M_2 for her and later creates a new message $M = M_1 \times M_2$, attacker can later claim that the sender has signed M. The attack is also referred to as multiplicative attack. This is a very serious attack on the RSA digital signature scheme because it is a selective forgery

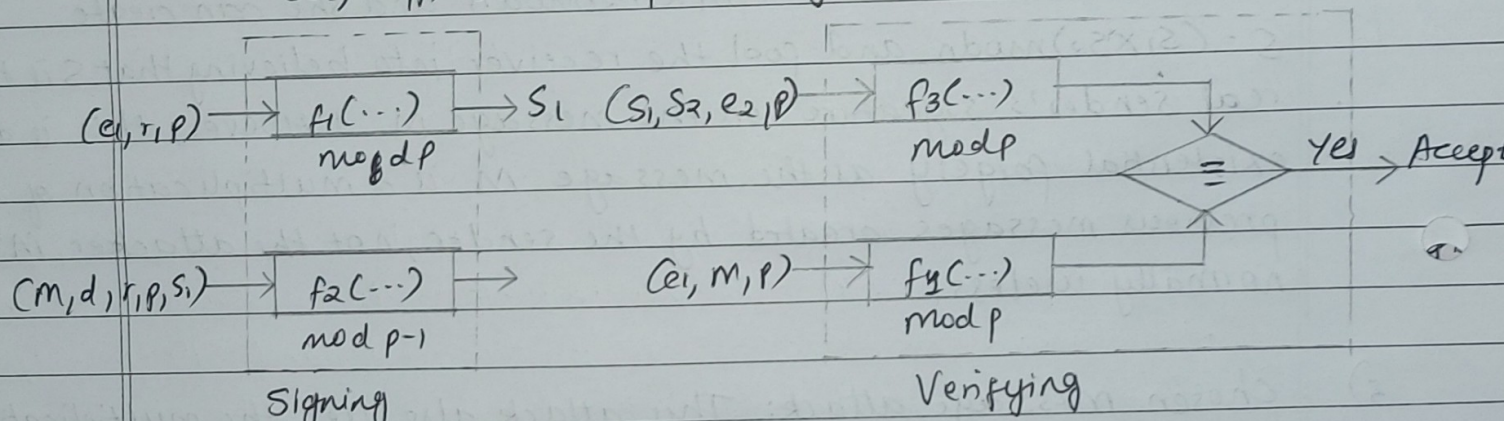
Q.2. Explain the El-Gamal Digital signature scheme along with forgery in the El Gamal digital signature.

⇒ The ElGamal digital signature scheme uses the same keys, but the algorithm, as expected, is different.

In the signing process, two functions create 2 signatures; in the verifying process, the outputs of the two functions are compared for verification. Note that one function is used both for signing and verifying but the function uses different outputs. The message is a part of the input function 2 when signing; it is a part of the input to function 1 when verifying. Note that the calculations in functions 1 and 3 are done modulo

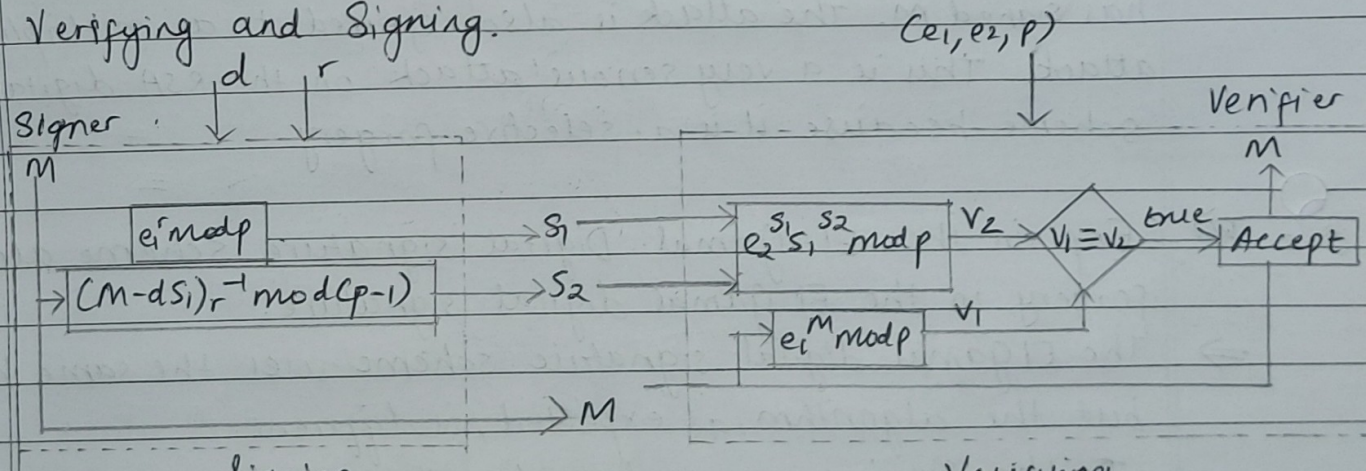
p: it is done modulo p-1 in function 2.

S_1, S_2: Signatures
M = message
(e_1, e_2, p): Sender's public key.
d = Sender's private key
r: random secret



General idea behind the El-Gamal signature scheme.

Verifying and Signing.



Signing: Signer can sign the digest of a message to any entity including the verifier:

a) Signer chooses a secret random no. r. Note that although public and private keys can be used repeatedly, the signer needs a new r each time she signs a new message.

(i) Signer calculates the first signature $S_1 = e_1^r \bmod p$

(ii) Signer calculates the second signature $S_2 = (M - d \times S_1) \times r^{-1} \bmod (p-1)$ where r^{-1} is the multiplicative inverse of r modulo p.

(iv) Signer sends M, S_1, S_2 to receiver.

Verifying: An entity such that the verifier receives M, S_1, S_2 which can be verified as follows:

(i) Verifier checks to see if $0 < S_1 < p$

(ii) Verifier checks to see if $0 < S_2 < p-1$

(iii) Verifier calculates $V_1 = e_1^M \bmod p$

(iv) Verifier calculates $V_2 = e_2^{S_1} \times S_1^{S_2} \bmod p$

(v) If V_1 is congruent to V_2, the message is accepted; otherwise it is rejected. We can prove the verification criterion using $e_2 = e_1^d$ and $S_1 = e_1^r$

$V_1 \equiv V_2 \pmod p \rightarrow e_1^M \equiv e_2^{S_1} \times S_1^{S_2} \pmod p \equiv (e_1^d)^{S_1} (e_1^r)^{S_2} \pmod p \equiv e_1^{dS_1 + rS_2} \pmod p$

We get $e_1^M \equiv e_1^{dS_1 + rS_2} \pmod p$

Forgeries in El-Gamal Digital signature scheme.

1) Key-only forgery: In this type of forgery, attacker has access to the public key. Two kinds of forgeries valid signatures S_1 and S_2 must be found by the attacker for this message. This is a selective forgery.

(a) Attacker can choose S_1 and calculate S_2. She needs to have $d^{S_1} S_1^{S_2} \equiv e_1^M \pmod p$. In other words, $S_1^{S_1} \equiv e_1^M d^{-S_2} \pmod p$ or $S_2 \equiv \log_{S_1} (e_1^M d^{-S_1}) \pmod p$. This means computing the discrete logarithm, which is very difficult.

(b) Attacker can choose S_2 and calculate S_1. This is much harder than part a.

c) Attacker may be able to find 3 random values M, S_1 and S_2 such that the last two are the signature of the first one. If the attacker can find two new parameters x and y such that $M = xS_2 \bmod (p-1)$ and $S_1 = -yS_2 \bmod (p-1)$, she can forge the message, but it might not be very useful for her. This is existential forgery.

d) Known-Message Forgery: If attacker has intercepted message M and the two signatures S_1 and S_2, she can find another message M', with the same pair of signatures S_1 and S_2. However, note that this is also an existential forgery that does not help the attacker very much.

Q.3. Discuss in details the process of how a message (digest) is digitally signed? Also discuss in detail various security provided by digital signature.

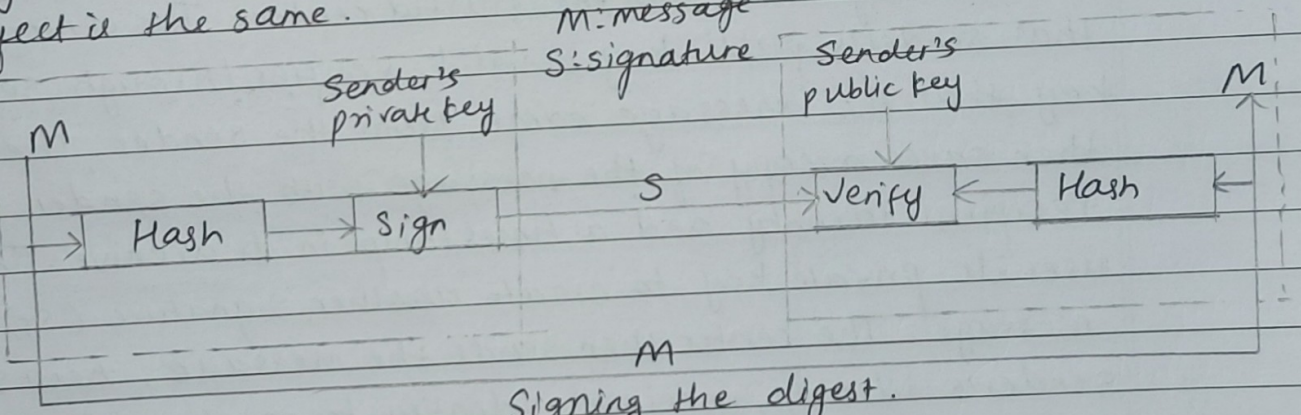
⇒ The sender uses a "signing algorithm" to sign the message. The message and the signature are sent to the receiver. The receiver receives the message and the signature and applies the verifying algorithm to the combination. If the result is true, the message is accepted, otherwise it is rejected.

Needs for keys

In a digital signature, the signer uses her private key, applied to a signing algorithm, to sign the document. We can add the private and public keys to give a more complete concept of digital signature. Note that when a document is signed, anyone, including the receiver can verify it because everyone has access to sender's public key. Sender must not use her public key to sign the document because then anyone could forge her signature.

Signing the Digest:

In digital signature system, the messages are normally long, but we have to use symmetric-key schemes. The solution is to sign a digest of the message, which is much shorter than the message. A carefully selected message has a one-to-one relationship with the message. The sender can sign the message digest and the receiver can verify the message digest. The effect is the same.



Signing the digest.

A digest is made out of sender's site. The digest then goes through signing process using sender's private key. Sender then sends the message and signature to the receiver. At the receiver's site, using the same public ~~key~~ hash function, a digest is first created out of the received message. Calculations are done on the signature and digest. The verifying process also applies criteria on the result of the calculation to determine the authenticity of the signature. If authentic, the message is accepted, else, rejected.

Security provided by digital signature

1) Message authentication: A secure digital signature, like a secure conventional signature can verify the authenticity of the message. Receiver can verify that a message is sent by the specific sender because that sender's public key is used in verification.

2) Message Integrity: The digital signature uses hash function in the signing and verifying algorithms that preserve the integrity of the message.

3) Nonrepudiation: People can create an established trusted party among themselves. Sender creates a signature from her message (S_A) and sends the message, her identity, the receiver's identity and the signature to the trusted center. The center, after checking that sender's public key is valid, verifies through sender's public key that the message came from the sender. The center then saves a copy of the message with the sender identity, recipient identity and a timestamp in its archive. The center uses its private key to create another signature (S_T) from the message. The center then sends the message, new signature, sender's identity, receiver's identity to receiver. Receiver verifies the message using the public key of the trusted center. In the future, if the sender denies that she sent the message, the center can show a copy of the saved message.

4) Confidentiality: A digital signature alone, cannot provide confidentiality. If confidentiality is required, the message and the signature must be enencpted using either a secret-key or a public-key cryptosystem.

Q-4. Compare and constrast following hashing algorithms.

(i) MD5 and SHA1

Criteria	MD5	SHA1
Security	Broken and insecure for cryptographic purposes. Vulnerable to collision attacks.	considered more secure and resistant to collision attacks, although it is also outdated for most cryptographic applications.
Length	MD5 produces a 128-bit hash value	SHA-1 produces a 160-bit hash value.
Speed	Faster	Slower.
Usage	MD5 was widely used for checksums and data integrity verification. However due to its vulnerabilities, it is no longer recommended.	SHA-1 has also been replaced by newer hash functions like SHA-256 for most purposes.

(ii) SHA-1 and SHA-256

Criteria	SHA-1	SHA-256
Output size	Produces a 160-bit hash value.	Produces a 256-bit hash value. This leads to stronger security against Brute force attacks.
Security	Considered insecure for many cryptographic purposes due to vulnerabilities that allow for collision attacks.	Considered secure and resistant to collision attacks.

Performance	Faster	Slower
Usage	Currently replaced by SHA-256	Used for a variety of cryptographic applications, including SSL/TLS certificates, cryptocurrency, and digital signatures.

(iii) SHA-256 and SHA-512.

Criteria	SHA-256	SHA-512
Output size	Produces a 256-bit hash value.	Produces a 512-bit hash value.
Performance	Faster than SHA-512 on 32-bit platforms as it operates on 32-bit words.	Faster on 64-bit platforms as it operates on 64-bit words.
Padding	Pads the message to a multiple of 512 bits	Pads the message to a multiple of 1024 bits.
Usage	Commonly used for a wide range of Cryptographic applications including SSL/TLS certificates, cryptocurrency and digital signatures.	Often used where a higher level of security is required

Q.5. What do you mean by digital certificate along with role of PKI in digital certificate?

⇒ A digital certificate, also known as a public key certificate or identity certificate, is an electronic document that verifies the ownership of a public key. It is issued by a certificate authority (CA) and contains information about the key, its owner and the CA that issued it. Digital certificates are used in many security protocols to authenticate the identity of users and devices, encrypt data and establish secure communications over the internet.

The Public Key Infrastructure (PKI) is a set of policies, procedures and technologies used to manage digital certificates and public-private key pairs. PKI plays a crucial role in the issuance, distribution and management of digital certificates. Some key roles of PKI in the context of digital certificates include:

1) Certificate issuance: PKI defines the process by which CAs issue digital certificates to users or devices. This process typically involves verifying the identity of the certificate requester and ensuring that the public key in the certificate is associated with that identity.

2) Certificate ~~distribution~~ Revocation: PKI provides mechanism for revoking digital certificates that are no longer valid. This is important to prevent the use of compromised or expired certificates.

3) Certificate distribution: PKI provides mechanism for securely distributing digital certificates to users and devices. This may involve using secure protocols such as HTTPS or email encryption.

4) Certificate Validation: PKI defines how digital certificates are validated by replying parties to ensure their authenticity and integrity. This often involves checking the certificate's digital signature and verifying its chain of trust back to a trusted root CA

5) Key Management: PKI includes procedures for managing public-private key pairs, including key generation, storage and backup. This helps ensure the security and integrity of digital certificates.

Q.6. Explain how Quantum Cryptography provides confidentiality to data.

⇒ Quantum cryptography provides confidentiality to data through the use of quantum principles to secure communication channels. The key idea behind quantum cryptography is the use of quantum properties to create a secure key exchange mechanism that is immune to eavesdropping.
One of the most well-known quantum cryptography protocols is the BB84 protocol developed by Charles Benett and Gillies Brassard in 1984. Here is a simplified explanation of how it works:

1) Quantum key Distribution (QKD): Alice wants to securely communicate with Bob. They both have a quantum communication channel and a classical communication channel.
2) Key Generation: Alice randomly prepares a sequence of qubits (quantum bits) in one of four possible states (0, 1, + or −). She sends these qubits to Bob over the quantum channel.
3) Measurement: Bob randomly measures each qubit he receives using one of two possible bases. The choice of basis determines whether he measures the qubit in the standard (computational) basis or in the Hadamard basis.
4) Error Detection: Alice and Bob communicate over the classical channel to compare which bases they each used for each qubit. If they used the same ~~qubits~~ basis, Bob's measurement results is random.
5) Key reconciliation: Alice and Bob discard the qubits for which they used different bases and agree on a subset of matching

bits. The subset forms their shared secret key.

6) Privacy Amplification: To further secure the key against potential eavesdropping, Alice and Bob apply a privacy amplification technique that reduces the amount of information an eavesdropper could learn about the key.

Q.7. Explain difference between classical cryptography and Quantum Cryptography.

⇒

Criteria	Classical Cryptography	Quantum Cryptography
1) Underlying Principles	Relies on mathematical algorithms and computational complexity assumptions. Ce.g. factoring large numbers, discrete logarithms) for security.	Relies on the principles of quantum mechanics such as properties of quantum states and quantum uncertainty to achieve security.
2) Key Generation	Uses classical algorithms to generate and distribute keys, which can be vulnerable to interception or brute-force attacks	Uses quantum principles (e.g. superposition, entanglement) to generate and distribute keys securely, ensuring that any eavesdropping attempts are detectable.
3) Key Exchange	Relies on a classical key exchange protocols (e.g. Diffie-Hellman, RSA) that are vulnerable to quantum attacks. (e.g Shor's algo for factoring large numbers)	Uses quantum key distribution (QKD) protocols for e.g (BB84, E91) that offer unconditional security, meaning they are theoritically secure against any computational attack, including quantum attacks.

4)	Security Model	Security is based on the computational complexity of breaking the cryptographic algorithm, assuring that the attackers computational resources are limited.	Security is based on laws of quantum mechanics, specifically the no-cloning theorem and the disturbance caused by measuring quantum states, which ensure that any eavesdropping attempts are detectable.
5)	Key Management	Requires secure key management practice to protect keys during generation, distribution and storage.	Offers the potential for more secure key generation and distribution, reducing the reliance on secure key management practices.